Tracking and Modeling People in Video Sequences

Ralf Plänkers and Pascal Fua [1]

Computer Graphics Lab (LIG)
Swiss Federal Institute of Technology
CH 1015 Lausanne (EPFL)

Tracking and modeling people from video sequences has become an increasingly important research topic, with applications including animation, surveillance and sports medicine. In this paper, we propose a model based 3–D approach to recovering both body shape and motion. It takes advantage of a sophisticated animation model to achieve both robustness and realism. Stereo sequences of people in motion serve as input to our system. From these, we extract a $2\frac{1}{2}$–D description of the scene and, optionally, silhouette edges. We propose an integrated framework to fit the model and to track the person's motion. The environment does not have to be engineered. We recover not only the motion but also a full animation model closely resembling the subject. We present results of our system on real sequences and we show the generic model adjusting to the person and following various kinds of motion.

Key Words: Shape, 3–D whole-body modeling and tracking, silhouettes

1. INTRODUCTION

Tracking and modeling people from video sequences has become an increasingly important research topic, with applications including animation, surveillance and sports medicine. In this paper, we propose a 3–D approach to recovering both body shape and motion. We obtain stereo- and silhouette-data from synchronized cameras and we fit to it a sophisticated body model. We use it to eliminate erroneous data, to robustly track complex motions even in the presence of occlusions and to derive a realistic body shape.

A detailed description of the human body in the form of an animated layered model is at the root of our work. It provides *a priori* information about the shape, and the allowable motions of the human body. This is essential for interpreting noisy data and solving the resulting ambiguities. The model we use is made of volumetric primitives attached to an articulated skeleton. Each one generates a potential field and the skin is taken to be an isosurface of the combined potential [32]. This implicit surface formulation has several

[1] The work reported here was funded in part by the Swiss National Science Foundation.

E-mail: {Ralf.Plaenkers, Pascal.Fua}@epfl.ch

advantages, among them a lower number of parameters and a 3–D distance measure that is differentiable and fast to compute.

As input to our system we use image sequences of people in motion, such as the one in Figure 1. Multiple synchronized and calibrated cameras are used to extract stereo information. Because cameras are relatively cheap and disparity maps such as the ones we use can be acquired at frame rate on ordinary computers [22], this is not a major limitation for many applications. Also, stereo works well both on textured clothes and on bare skin. Silhouette edges can be included when available. Stereo and silhouettes are complementary sources of information: Stereo works well where the surface faces the camera but fails where the surface slants away. Silhouettes, on the other hand, provide information exactly there, at the occluding contour.

We have developed an extensible least squares framework that we use to fit the body model to the different types of input data, with minimal human intervention. To initialize the process, the user simply clicks on the approximate location of a few key-points in one image pair. The recovered shape and motion parameters can then be used to reconstruct the original motion, to display it from a different viewpoint or to make other animation models mimic the subject's actions.

To overcome the problems inherent to least-squares optimization, we introduce two weighting schemes. The first ensures that diverse information sources, such as stereo and silhouettes, have commensurate influences so that they can be combined. The second accounts for the fact that more data may be available for some body parts than for others. It prevents the system from exclusively fitting the former at the expense of the latter.

Recently, techniques have been proposed [15, 23, 6] to track human motions from video sequences. They are fairly effective but use very simplified models of the human body, such as ellipsoids or cylinders, that do not precisely model the human shape. The recovered motion can indeed be applied to other models. However, a model of the filmed person that would be sufficient for a truly realistic animation is not obtained. The interested reader is referred to the recent surveys in [14, 24] for further references.

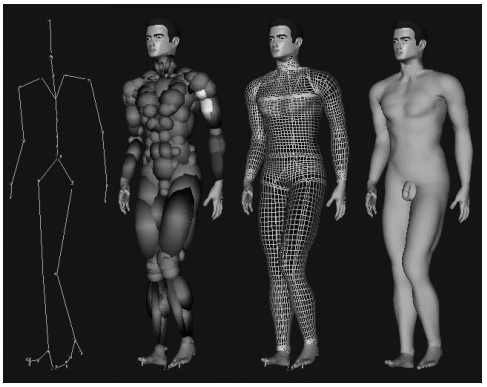
Other very promising approaches to tracking are probability-based. The Kalman filter framework [16] and, more recently, the CONDENSATION [19, 2] algorithm have become popular. Again, these approaches only cover the tracking aspect—albeit very well—and tend to neglect the modeling part which this paper emphasizes. Thus, probability-based algorithms could be used to drive the tracking part and our optimization-based algorithm could drive the modeling part.

Much work has also been devoted to the use of silhouettes for body modeling [20, 8, 18, 5, 9]. They provide very useful but incomplete information about shape which is one of the issues we address in this work. The main limitation of 2–D approaches is the presence of occlusions. Thus, several authors require a set-up of at least three orthogonal cameras. We also use several cameras to compute stereo but they do not have to be positioned precisely. This greatly simplifies the setup of our system. Furthermore, our 3–D models allow us to handle occlusions.

While laser scanning technology provides a fairly good surface description of a static object [30], using video sequences allows us in addition to measure and track the person in motion and, thus, to recover the positions of the articulations inside the skin surface. Also, full body scanners are expensive compared with standard video cameras.

Motion capture systems address the problem of accurately tracking human motion. Magnetic or optical markers are attached on the limbs of the person and their 3–D trajectories are

FIG. 1. Original sequence of an upper body motion. Frames 20, 40, 60 and 80 out of 100 from one camera are shown.



(a) (b) (c) (d)

FIG. 2. The layered human body model: (a) Skeleton. (b) Ellipsoidal metaballs used to simulate muscles and fat tissue. (c) Polygonal surface representation of the skin. (d) Shaded rendering.

used to infer the subject's motion. These systems are mostly automatic but are cumbersome to use and too expensive for mass use or low budget projects. Furthermore, even with highly professional systems, measurement errors and ambiguities in the automatic matching procedures result in tracking errors and, thus, manual intervention is needed. Several of the techniques presented here can also be used to overcome some of those limitations [28, 17].

In the remainder of the paper, we first introduce the body animation model we use. Then, we describe how we fit this model to the data. We present our fitting procedure and our approach to handling the different kinds of input information. Finally, we present reconstruction results on complex human motions.

2. MODELS

2.1. The Layered Approach

The human body model we use in this work [32] is depicted by Figure 2. It incorporates a highly effective multi-layered approach for constructing and animating realistic human bodies. The first layer is a skeleton that is a connected set of segments, corresponding to limbs and joints. A joint is the intersection of two segments, which means it is a skeleton point where the limb linked to that point may move.

Smooth Implicit surfaces, so-called ellipsoidal *metaballs*, form the second layer. We will present *metaballs* in more detail in Section 2.3. They are used to simulate the gross behavior of bone, muscle, and fat tissue; they are attached to the skeleton and arranged in an anatomically-based approximation. The third layer, a polygonal skin surface, is constructed via B-spline patches over control points generated by a ray casting method [27, 32].

The key advantage of the layered methodology is that once the layered character is constructed, only the underlying skeleton need be scripted for animation; consistent yet expressive shape deformations are generated automatically.

2.2. Skeleton and State Vector

The state of the skeleton is described by the combined state vector

$$S^{body} = [S^{motion}, S^{skel}] \ . \tag{1}$$

Since the skeleton is modeled in a hierarchical manner, we can define the *static* or *init* state of the skeleton S^{skel} as the rotations and translations from each joint with respect to the preceding one. It is fixed for a given instance of the body model. The variable or *motion* state vector S^{motion} contains the actual values for each degree of freedom (DoF), i.e. the angle around the z-axis towards the next DoF. They reflect the position and posture of the body with respect to its rest position. All joints have a single angular DoF. More complicated articulations are split into several, single-DoF joints sharing the same location and only differing in their orientations.

The position of joints in a global or world referential is obtained by multiplying the local coordinates by a transformation matrix. This matrix is computed recursively by multiplying all the transformation matrices that correspond to the preceding joints in the body hierarchy:

$$X^j = \prod_{\kappa} D_\kappa(S) * X^w \ , \tag{2}$$

with $X^{j,w} = [x, y, z]^T$ being joint local, resp. world global, coordinates and the homogeneous transformation matrices D_κ, which depend on the state vector S, ranging from the root articulation's first to the reference articulation's last DoF. These matrices are split into static and motion matrices, according to the state vector. They are of the form

$$D = D^{rot_z} * D^{ini} \ . \tag{3}$$

The rotation matrix D^{rot_z} is defined by the motion state vector. It is a sparse matrix allowing only a rotation around the local z-axis (Θ_κ). The static transformation $D^{ini} = (RX + sT)$ is a matrix directly taken from the standard skeleton. These matrices translate by the bone length and rotate the local coordinate system from the joint to its parent. The matrix entries are calculated using values s from the state vector S^{skel}. The variable coefficient s is necessary because the exact size of the limbs may vary from person to person.

2.3. Metaballs and their Mathematical Description

2.3.1. Definition.

In Blinn's basic formulation [4], metaballs or *blobs* are defined by a set of points $P_i(x, y, z)$ that are the sources of a potential field. Each source is defined by a *field function* $F_i(x, y, z)$ that maps $\mathbb{R}^3$ to $\mathbb{R}$, or a subset of $\mathbb{R}$. At a given point $X(x, y, z)$ of the Euclidean space, the fields of all sources are computed and added together, leading to the global field function $F(x, y, z) = \sum_{i=1}^{n} F_i(x, y, z)$, with n being the number of sources. A curved surface can then be defined from the global field function F by giving a threshold value T and rendering the following equipotential surface $\mathbf{S}$ for this threshold:

$$\mathbf{S} = \left\{(x, y, z) \in \mathbb{R}^3 \ \mid \ F(x, y, z) = T\right\} \ . \tag{4}$$

Conceptually it is usually simpler to consider field function F_i as the composition of two functions [3]: the *distance function* d_i which maps $\mathbb{R}^3$ to $\mathbb{R}^+$, and the *potential function* f_i

which maps $\mathbb{R}^+$ to $\mathbb{R}$:

$$F(x,y,z) = \sum_{i=1}^{n} f_i(d_i(x,y,z)) \ . \tag{5}$$

The function $f_i(d)$ characterizes the distance between a given point $X(x,y,z)$ and the source point $P_i(x,y,z)$. Typically d_i is defined as a function of a user-provided parameter $r_a \in \mathbb{R}^+$ (called *effective radius*) which expresses the growing speed of the distance function. The most obvious solution for $d_i(x,y,z)$ is the Euclidean distance, but several other functions have been proposed in the literature, especially when the potential source is not reduced to a single point or its field is not equally distributed in space.

2.3.2. Distance function.

In this work, we only consider ellipsoids as primitives because they are relatively simple but, nevertheless, allow modeling of human limbs with a fairly low number of primitives and thus number of parameters. We represent the distance function d_i by the implicit distance to the ellipsoid that is

$$d_i(x,y,z) = \left(\frac{x}{lx}\right)^2 + \left(\frac{y}{ly}\right)^2 + \left(\frac{z}{lz}\right)^2 \ , \tag{6}$$

where $L_i = (lx, ly, lz)$ are the radii of the ellipsoid, i.e. half the axis length along the principal directions.

2.3.3. Potential Function.

The field value at any point X in space is defined by the distances between X and the source points P_i. The center of the primitive, its source, has the greatest density. The value of the primitive's density, decreases toward the element's outer edge, or effective radius. The visible size of a primitive, called the *threshold radius*, is determined by the effective radius and weight. To achieve visually pleasing results field functions should satisfy two criteria:

1. Extremum: The contribution at the source is some maximum value, and the field will drop smoothly to zero at a distance r_a, the effective radius.

2. Smoothness: In order to blend multiple metaballs smoothly and gradually, $f'(0) = f'(r_a) = 0$.

A single, lower degree polynomial cannot meet both criteria, hence either piecewise quadric or high order polynomials have been proposed. They tend to be complex and, thus, to imply high computational cost. In the original body modeling work [27], a simple quadric field function was used. It does not satisfy the above mentioned criteria, however, it comes close enough to yield visually nice results at a very low computational cost.

2.3.4. Special Potential Function

To permit an effective fit of our implicit surface model to the data, the field function must be differentiable at least over the range $[0..r_a]$ and it should drop smoothly towards zero. We therefore cannot use a simple quadratic field function. We chose to use an exponential

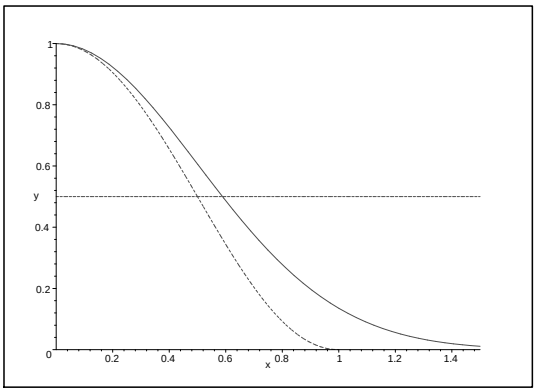


FIG. 3. One-dimensional plot of the exponential field function of Eq. 7. The horizontal axis represents the distance from the source and the vertical axis plots the corresponding field value. The threshold $T = 0.5$ is also depicted. The ellipsoidal kernel of this metaball (d_i) has an effective radius (r_a) of 1. The dashed curve depicts a classical piecewise quadratic function, i.e. it fulfills both criteria of Section 2.3.3.

field function instead:

$$f_i = w_i \left(\frac{1}{e^{d_i}} \right)^2 = w_i * exp(-2d_i) \quad , \tag{7}$$

with d_i being the distance of Equation 6 and weight and threshold being fixed for the moment ($w_i = 1$, $T = 0.5$). In the future, we might leave the weight as a free parameter for the fitting because it allows the modeling of sharper edges. Figure 3 depicts a plot of this special potential function in one dimension compared to a classical piecewise quadratic function.

The equipotential surface $\mathbf{S}$ of an exponential field function is only slightly different from the standard representation [4] and, more importantly, it never falls to zero as depicted by Figure 3.

This last property has the following consequence: Each blob has an influence on all other blobs of the same limb, although, it will become very small for distant blobs. This is undesirable for modeling purposes since the designer looses local control but favorable for automated tracking or fitting purposes. At the same time as each blob influences all other blobs, each blob is influenced by all observations in our fitting framework. This allows us to work with only a rough initialization of the model's posture because of the long range effect of the exponential field function.

2.4. Usage for Tracking and Modeling

The human body model was initially developed for animation purposes. It has been successfully used to produce and animate very realistic models. In the following section, we will show that it can also be used to track and to fit by adjusting a relatively small number of parameters.

In this work we use models of different levels of detail, that is a simple model with only few metaballs (54) to speed up tracking and a more complex one with about 230 metaballs for shape modeling. To further reduce the number of parameters we introduced higher level parameters that control groups of metaballs. For example "upper arm width" which controls the relative size of all metaballs in the upper arm region.

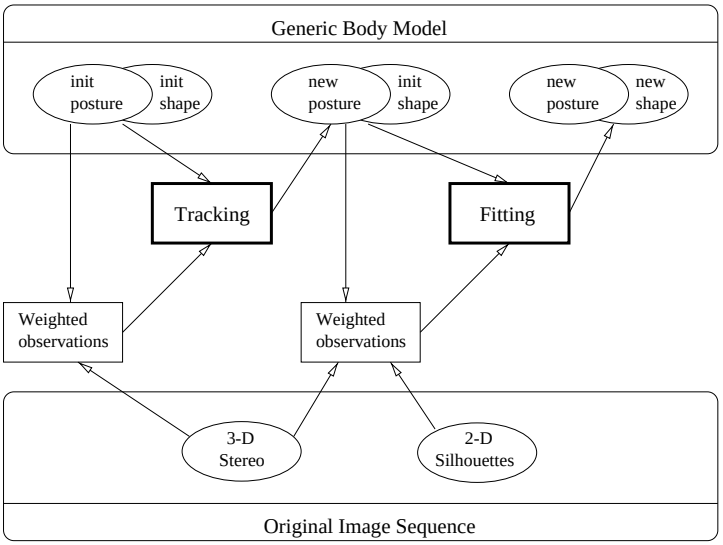


FIG. 4. Flow chart of our system.

3. FITTING THE MODELS TO IMAGE DATA

From a fitting point of view, the body model of Section 2 embodies a rough knowledge about the shape of the body and can be used to constrain the search space. Our goal is to derive its degrees of freedom so that it conforms as faithfully as possible to the image data.

Here we use motion sequences such as the ones shown in Figures 9, 10 and 11. Figure 5 depicts the corresponding stereo data. Silhouette information can be added when available, as shown in Figure 7. Thus, the expected output of our system is a state vector that describes the shape of the metaballs and a set of joint angles corresponding to their positions in each frame.

3.1. Approach outline

Figure 4 depicts the flow of our algorithm. The body model and the image data are used throughout the process. The algorithm works as follows:

Data Acquisition.

Clouds of 3–D points are derived from the input images. Silhouette edges may be delineated in several key-frames or be automatically generated for the whole sequence.

Initialization.

We first initialize the model interactively in one frame of the sequence. The user has to enter the approximate position of some key joints, like shoulders, elbows, hands, hips, knees and feet. Here, it was done by clicking on these features in two images and triangulating the corresponding points. Alternatively we could use anthropometry-based methods [31, 1] to initialize them with a few clicks in one single image. This initialization gives us a rough shape, i.e. a scaling of the skeleton, and an approximate model posture.

Tracking.

At a given time step the *tracking* process adjusts the model's joint angles by minimizing an objective function that will be described in Section 3.2. This modified posture is saved for the current frame and serves as initialization for the next one. The computing power of

todays PCs allows for interactivity. If, for some reason, the algorithm loses track the user simply pauses the program, adjusts the posture interactively and hands the control back to the algorithm for further processing.

Fitting.

The results from the *tracking* step serve as initialization for a *fitting* step. Its goal is to refine the postures in all frames and to adjust the skeleton and/or metaball parameters to make the model correspond more closely to the person. The *fitting* optimizes over all frames simultaneously, again by minimizing the objective function described in Section 3.2. This allows us to find a single set of parameters that describes a model that is consistent with the images of the whole sequence. The results could be further improved by introducing inter-frame constraints such as smoothness or limits on velocity/acceleration. This will be the object of future work.

The purpose of the simultaneous fitting is the following: In order to correctly model the proportions of the skeleton, i.e. the exact position of the articulations inside the skin surface, we need to observe the person in motion and find a configuration which conforms to every posture.

Results.

The results of the *fitting* are a new set of skeleton and primitive parameters S^{skel} and a sequence of motion parameters S^{motion} that make the recovered model mimic the subject's action.

Both the *tracking* and the *fitting* step use the same algorithm, a least squares optimizer. The following paragraph describes the various steps of this framework in more detail.

3.2. Least Squares Framework

In standard least-squares fashion, we use the image data to write $nobs$ observation equations of the form

$$y_i(S) = obs_i - \epsilon_i \quad , 1 \leq i \leq nobs \quad , \tag{8}$$

where S is the state vector of Equation 1 that defines the shape and position of the limb and ϵ_i is the deviation from the model. We will then minimize

$$v^T P v \Rightarrow Min \ , \tag{9}$$

where $v = [\epsilon_1, \ldots, \epsilon_{nobs}]$ is the vector of residuals and P is a weight matrix associated with the observations. P is usually introduced as diagonal.

Our system must be able to deal with observations coming from different sources that may not be commensurate with each other. Formally we can rewrite the observation equations of Equation 8 as

$$y_i^{type}(S) = obs_i^{type} - \epsilon_i \quad , 1 \leq i \leq nobs \quad , \tag{10}$$

with weight p_i^{type}, where $type$ is one of the possible types of observations we use. In this paper, $type$ can be object space coordinates or silhouette rays. However, other information cues can easily be integrated.

FIG. 5. An original image pair and the corresponding disparity map.

The individual weights of the different types of observations have to be homogenized prior to estimation according to:

$$\frac{p_i^k}{p_j^l} = \frac{\left(\sigma_j^l\right)^2}{\left(\sigma_i^k\right)^2} \ , \tag{11}$$

where σ_j^l, σ_i^k are the a priori standard deviations of the observations obs_i, obs_j of type k, l.

Least-squares estimation means finding the joint minimum

$$\sum_{type=1}^{nt} v^{type} P_{type} v^{type} \Rightarrow Min \ , \tag{12}$$

where nt is the number of observation types. It yields the well-known normal equations which need to be solved using standard techniques.

In practice, however, it is very difficult to estimate the standard deviations of Eq. 11. We therefore use the following heuristics which has proved to be very effective. To ensure that the minimization proceeds smoothly we multiply the weight p_i^{type} of the n_{type} individual observations of a given type by a global coefficient c_{type} computed as follows:

$$\begin{aligned} G_{type} &= \frac{\sqrt{\sum_{1 \le i \le nobs, j = type} p_i^{type} \|\nabla f_i^j(S)\|^2}}{n_{type}} \\ c_{type} &= \frac{\lambda_{type}}{G_{type}} \end{aligned} \tag{13}$$

where λ_{type} is a user supplied coefficient between 0 and 1 that indicates the relative importance of the various kinds of observations. This guarantees that, initially at least, the magnitudes of the gradient terms for the various types have the appropriate relative values.

Since our overall problem is non-linear, the results are obtained through an iteration process. We use an implementation of the Levenberg-Marquardt algorithm [26] that can handle the large number of parameters and observations we must deal with.

3.3. Using Stereo Data

In this work we used a single correlation-based based stereo algorithm [11] to compute dense disparity maps from two or more cameras. It produces disparity maps such as the one shown in Figure 5. Calibrating the cameras allows us to generate clouds of 3–D points from the depth maps. We want to minimize the distance of the reconstructed limb to all such "attractor" points. Given the implicit description of the metaballs of Eq. 7, the simplest way to achieve this result is to write a pseudo-observation equation of the form:

$$\sum_{i=1}^{np} w_i \cdot e^{-2d_i(X)} \;=\; T - \epsilon \tag{14}$$

$$\sum_{i=1}^{np} \left(\frac{1}{e^{\left(\frac{x_i}{lx_i}\right)^2 + \left(\frac{y_i}{ly_i}\right)^2 + \left(\frac{z_i}{lz_i}\right)^2}} \right)^2 \;=\; \frac{1}{2} - \epsilon \;, \tag{15}$$

where np is the number of primitives for this body part, $X\,(x_i, y_i, z_i)$ is the 3–D observation transformed into the local coordinates of primitive i with radii $L_i(lx, ly, lz)$. We use Equation 15 which is the same as Equation 14 except for the fixed weights $T = 0.5$, $w_i = 1$ and the substitution according to Eq. 6.

The optimization is effected wrt. the primitives' radii L_i and the DoFs which reside in the transformation of each observation from world global to primitive local coordinates. These DoFs consist of the motion parameters and the skeleton parameters, i.e. length of each limb. According to Equation 2, each observation X can be written as a function of its world coordinates and the elements of state Vector S.

3.4. From Silhouette Data to Observations

Contrary to 3–D edges, silhouette edges are typically 2–D features since they depend on the viewpoint and cannot be matched across images. However, they constrain the surface tangent. Each point of the silhouette edge defines a line, the camera ray, that goes through the optical center of the camera and is tangent to the surface at its point of contact with the surface. The points of a silhouette edge therefore define a ruled surface that is tangent to the surface to be modeled.

In terms of our model fitting, this means that exactly one point of the silhouette ray must lie on the metaball and its normal must be orthogonal to the silhouette ray. This can be expressed as follows:

$$\sum_{i=1}^{np} w_i \cdot e^{-2d_i(X)} \;=\; T - \epsilon \tag{16}$$

$$Slope \cdot \left[\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}, \frac{\partial f}{\partial z} \right] \;=\; 0 - \epsilon \tag{17}$$

where $Slope$ is the tangent's direction and $d_i(X)$ is the distance of the silhouette ray to the metaball (Eq. 6) with X being the point on the ray that is closest to the metaball. These two constraints are depicted by Figure 6.

3–D position of silhouette edges.

The main difficulty is to find the metaball surface point X where the constraint applies. In practice, we take this point to be the point on the camera ray which minimizes the implicit formulation of the model, Eq. 7.

This is a reasonable approximation when the initial position of the model is not too far from the real one. This particular choice has one further advantage: It allows us to ignore the constraint of Eq 17 for the following reason.

After each iteration of the least-squares optimizer we reestimate the position of the silhouette edge using the current surface model. This is both necessary and desirable. It is

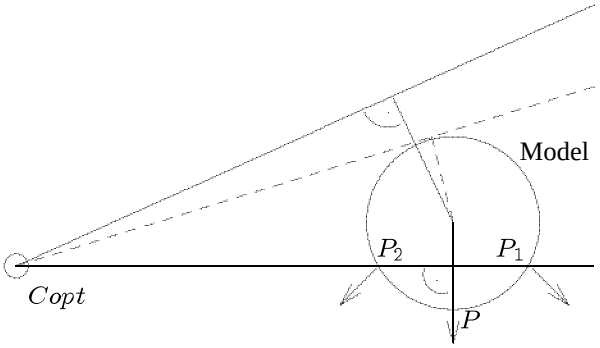


FIG. 6. A 2–D silhouette ray for a circular object ($Model$) is represented by a dashed line. The camera is depicted by $Copt$. The ray touches the circle at exactly one point and it's slope is orthogonal to the object's normal in the point of intersection. The two other rays don't satisfy both of the silhouette criteria of Section 3.4.

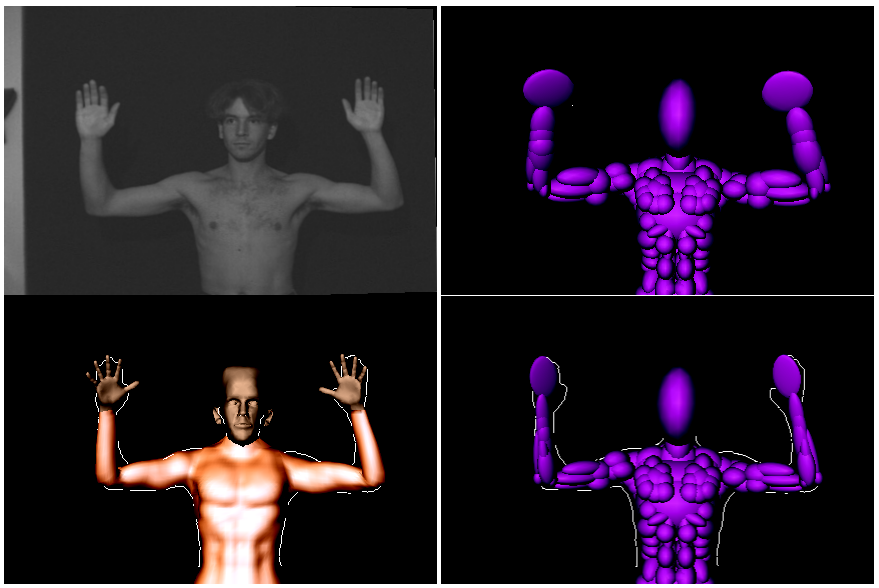


FIG. 7. The importance of silhouette information for shape modeling. The original image is shown in the upper left. In the upper right no silhouette constraints were used and the fitting puts the model is too far away from the cloud. This is compensated by enlarging the primitives. The silhouettes provide stricter constraints for the model. The lower row shows the result of the fitting with and without skin rendered.

necessary because, after an iteration, the model has changed and the point of the camera ray that is closest to the model is likely to have moved. It is desirable because, after this re-initialization of the silhouette edge, this point will be where the silhouette tangent is closest to the model and will thus satisfy the orthogonal normal constraint.

Informally, this can be verified as follows: our models have an ellipsoidal shape and are expressed in an implicit manner. The closest point of a ray can be found by looking for the smallest isosurface which still touches the ray, whether outside or inside the original model. These isosurfaces also are ellipsoids with the same center as the model. Furthermore, a ray that touches an ellipsoid at exactly one point is tangent to this ellipsoid and thus, the normal of the model at that point is orthogonal to the ray. An alternative way to handle these constraints is presented in [29] and we intend to implement their method in the future and compare these two approaches.

FIG. 8. Automatically tracked silhouette of a walking person.

The importance of using silhouette information is demonstrated by Figure 7. Here, we allowed for changes in the model's posture and the shape parameters of the arms. In the upper row of Figure 7 only the 3–D information is used. The fitting tends to move the model further away from the cloud and to compensate by inflating the arms to keep contact with the point cloud. The noisy stereo data is too ambiguous to sufficiently constrain the model. The silhouettes are needed to constrain it, as shown in the lower row of Figure 7 where we fitted to both stereo and silhouette information.

Obtaining silhouette information.

This can be achieved in many ways. Many authors cite the use of Canny edge detectors in images with subtracted background. This is an automatic but low-level method and thus relatively easy to implement but not very robust in practice. Automated silhouette edge detectors have been developed and could be implemented for this use [33]. In this work, we have used semi-automated tools to allow the user to quickly sketch the silhouette edges [25].

We also experimented with a snake-based silhouette tracker that uses 3–D point trajectory information. The silhouette is interactively initialized in the first frame, as shown in Figure 8. The system is then able to track it over the whole sequence, in spite of the highly cluttered and dynamic background. Note that another person is walking in the background. Correlating not only the two images of a stereo pair but also succeeding images acquired with the same camera provides sparse, yet quite robust 3–D trajectories for textured surfaces. [7] The stereo depth information is used to extract the foreground and, where available, the temporal tracks are used to predict silhouette motion and to constrain the snake's optimization. For more details, we refer the interested reader to our technical report [13].

3.5. Motion Prediction

In order to increase robustness of the tracking algorithm, we introduced a constant-velocity prediction model. We first tried a simple linear extrapolation from the two preceding time steps. In practice, this simplistic approach is not usable due to the noisiness of the data. Small errors in the tracking are immediately extrapolated and, thus, exaggerated. Often, this results in losing track even in the absence of occlusions. To circumvent this problem we introduced a smoothed prediction over the previous n frames; here we used $n = 5$. Again, a constant velocity model is used to extrapolate from the previous frames. In future work we intend to implement a method that compares predicted and unpredicted model state and picks the one that yields the smallest residual. Also, more sophisticated motion models or learned motions could be used.

3.6. Data Segmentation

Segmenting data is still a challenging problem in the field of computer vision. In our case, we need to decide which part of the body an observation should be assigned to. Robust image segmentation is not possible without forcing the performer to wear clothes of a specific texture or colour pattern. In order to deal with arbitrary types of images we propose a model-driven approach. The segmentation is done with respect to the current posture of the model. An observation is simply assigned to the closest body part. This is a hard assignment which is not guaranteed to be correct and a weighted or fuzzy assignment [21] might be used instead. In practice, we get good results with the segmentation being recomputed after each refinement of the model.

3.7. Dynamic Model Based Weighting

Due to the nature of least squares, we encountered problems dealing with the non uniform distribution of 3–D observations. The system tends to fit better those parts of the body where high number of observations are available, neglecting those with only little information. To overcome this problem we have introduced the following weighting scheme.

The number of 3–D points we can observe for a certain body part depends on the visible surface and the quality of texture. The following factors contribute to the size of the visible surface:

1. Absolute surface.
2. Exposure, i.e. angle wrt. cameras.
3. Visibility due to occlusion or field of view.

The obvious thing to do would be to precompute a weight factor based on each body part's absolute surface. But this is not sufficient, the other factors which make up the visible surface are too important to be ignored. And, anyway, we would still need to segment the observations in some way to attribute the different weights. This necessary segmentation can easily be further exploited.

In our system, we opted for a dynamic model based weighting scheme. The actual formula to compute the weight is very simple. And it implicitly covers all of the three factors mentioned above. With nobs being the number of all observations in this frame, ρ the number of separate body parts and ω the number of observations which have been attributed to the body part bp we compute the weight P as follows:

$$P_{bp} = \frac{\mathrm{nobs}}{\rho * \omega_{bp}}$$

We multiply this weight with p_i^{type} of Eq. 10. This conserves the homogenized weights because the sum of the weighted observations for all body parts equals the number of all observations: $\sum P_{bp} * \omega_{bp} = nobs$. Since the scheme is model-based and the model may change between iterations, we recompute the weights after each iteration.

3.8. Results

Human walking.

Our first test sequence consists of somebody walking in front of a horizontally aligned stereo camera pair. The background and lighting was uncontrolled (standard office head lights) and the camera pair was about 5m from the person. The distance between the

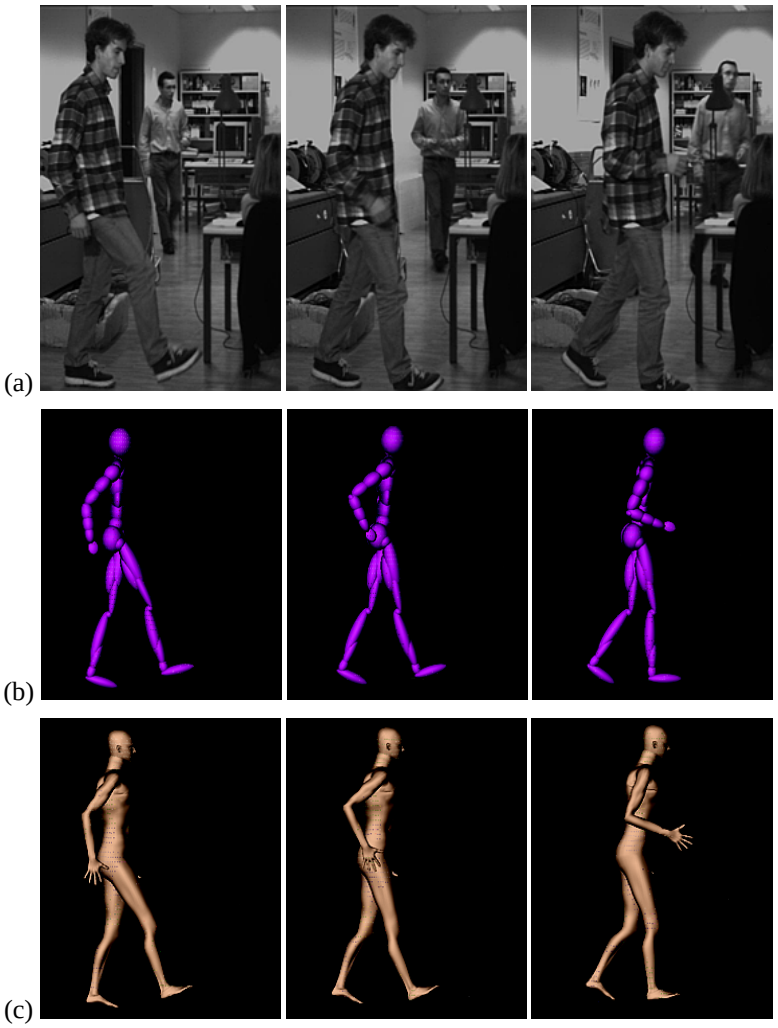


FIG. 9. Three frames of the walking sequence are shown. The original sequence is in the top row (a). The middle row (b) shows the tracking with the simple model, overlaid with the 3–D points. And the final fitting of the detailed model is shown in the bottom row (c).

two cameras was 75cm. The images are interlaced and the processed half-frame has an effective resolution of 768×288. The disparities result in about 2000 3–D points, including reconstructed parts of the background.

Figure 9(a) shows three frames (cropped) out of 50 from this sequence. The result from the initial tracking process is depicted by Figure 9(b). We had to manually interact when the legs crossed during the walking cycle because we didn't use any prediction techniques for this sequence. This would be needed to make the leg swing through when no data is available because of the occlusion.

The results of the subsequent fitting step are shown in Figure 9(c). Here, the dimensions of the skeleton and the size of the metaballs have been adjusted, resulting in a better model and slightly more realistic postures.

Upper body motion.

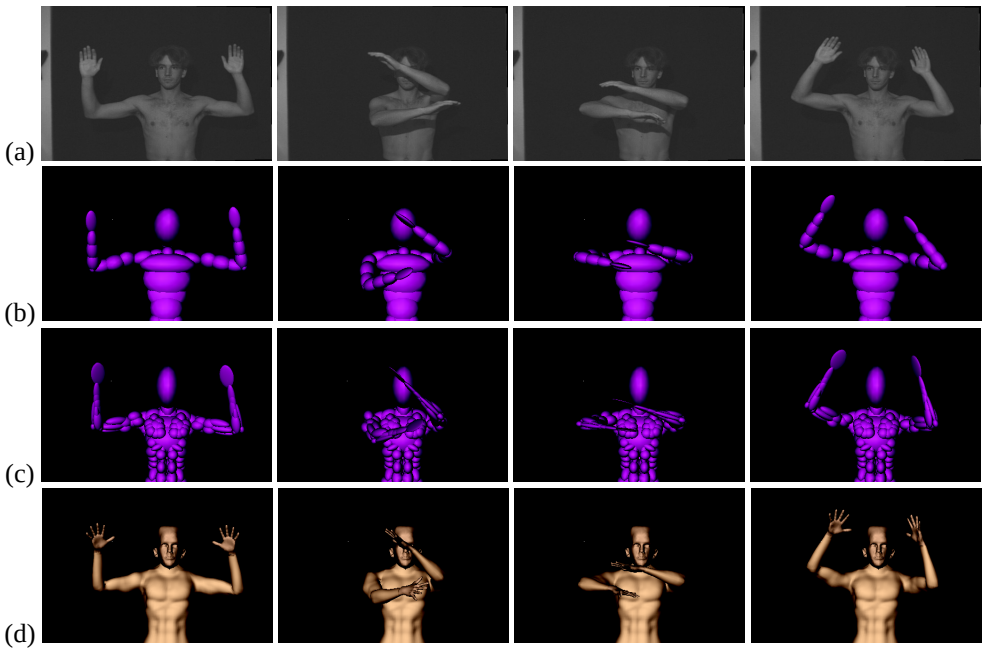


FIG. 10. In the top row (a) is the original sequence of the upper body motion. Frames 10, 50, 60 and 90 out of 100 are shown. Results of the tracking using the simple model are shown in (b) and the results of the fitting with the full animation model are shown in the bottom rows without (c) and with (d) skin rendered.

The sequence in Figure 10(a) shows complex movements of a naked upper body, taken with a camera set up in front of the subject. Three cameras in an L configuration were taking interlaced images at 20 frames/sec with an effective resolution of 432×288 per half-frame. Our stereo algorithm [10] produced very dense point clouds with about 4000 3–D points on the surface of the subject, even without textured clothes. To increase the frame rate and, thus, reduce the difference in posture between frames we used both halves of the interlaced images and adjusted the camera calibration accordingly.

The result of the tracking process is shown in Figure 10(b). The fitting step, using a more detailed model, produced slightly better postures, an adapted skeleton and resized metaballs (Fig. 10(c)). The head of this model was generated from a single video sequence of the subject by using the system of [12].

Figure 11(a) shows five frames of an upper body motion of a person with very different body proportions. The results are presented in Figure 11(b) and (c). They show that the generic model adjusts well to the new person who has very different body proportions compared to the previous example. The fitting had to adjust six angular degrees of freedom: elbow (1 DoF), shoulder (3 DoF) and torso (2 DoF) plus the skeleton parameters “lower arm length”, “upper arm length” and “shoulder width”.

4. CONCLUSION AND FUTURE WORK

We have presented a technique for fitting a complete animation model to image data and tracking complex 3–D motions. The model and the constraints it imposes are used to overcome the inherent noisiness of the data. We recover both motion and body shape from stereo video sequences. The corresponding parameters can be used to recreate realistic 3–D animations. Such a capability should be of great use in the area of human animation

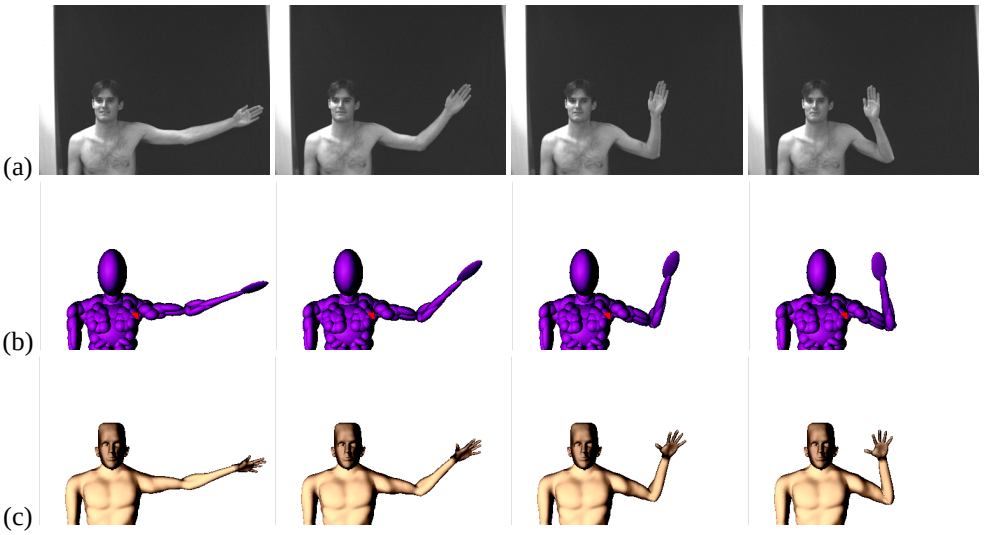


FIG. 11. Sequence of an upper body motion of another person. Frames 10, 30, 50 and 70 out of 100 are shown. The original sequence is in the top row (a), the fitting results are in the center row (b) and the final rendering of the skin surface is at the bottom (c).

since it could also be used to analyze and visualize human motion for medical and training purposes.

In future work, we intend to further exploit our strong model, for example the model can help to identify occlusions and decide whether to let the data guide the fitting or to let the prediction change the posture where no data is available. The model could also be used to derive an automatic and robust silhouette extraction algorithm, even with cluttered background.

Also, now, we search for the "optimal" skeleton and metaball parameters over the whole sequence. This is a very time consuming process and it could be sped up by identifying the *most interesting* frames and, then, fit only in these frames. *Most interesting* frames in this context could be frames where the model is close to some pre-defined postures which are known to have a high information content.

REFERENCES

1. C. Barron and I. Kakadiaris. Estimating anthropometry and pose from a single image. *Computer Vision and Image Understanding,* 2001. to appear.
2. A. Blake, B. Bascle, M. Isard, and J. MacCormick. Statistical models of visual shape and motion. *Phil. Trans. R. Soc. Lond.*, A(356):1283–1302, 1998.
3. C. Blanc and C. Schlick. Extended field functions for soft objects. In *Eurographics Workshop on Implicit Surfaces 95*, pages 21–32, Grenoble, France, 1995.
4. J. F. Blinn. A generalization of algebraic surface drawing. *ACM Transactions on Graphics*, 1(3):235–256, 1982.
5. M. Brand. Shadow Puppetry. In *International Conference on Computer Vision*, pages 1237–1244, Corfu, Greece, September 1999.
6. Ch. Bregler and J. Malik. Tracking people with twists and exponential maps. *Conference on Computer Vision and Pattern Recognition,* June 1998.
7. N. D'Apuzzo, A. Gruen, R. Plänkers, and P. Fua. Least Squares Matching Tracking Algorithm for Human Body Modeling. In *XIX ISPRS Congress*, Amsterdam, July 2000.
8. J.W. Davis and A.F. Bobick. A Robust Human-Silhouette Extraction Technique for Interactive Virtual Environments. In *Workshop on Modelling and Motion Capture Techniques for Virtual Environments*, pages 12–25, Geneva, Switzerland, November 1998.

9. Q. Delamarre and O. Faugeras. 3D Articulated Models and Multi-View Tracking with Silhouettes. In *International Conference on Computer Vision*, pages 716–721, Corfu, Greece, September 1999.
10. P. Fua. Reconstructing Complex Surfaces from Multiple Stereo Views. In *International Conference on Computer Vision*, pages 1078–1085, Cambridge, MA, June 1995. Also available as Tech Note 550, Artificial Intelligence Center, SRI International.
11. P. Fua. From Multiple Stereo Views to Multiple 3–D Surfaces. *International Journal of Computer Vision*, 24(1):19–35, August 1997.
12. P. Fua. Using Model-Driven Bundle-Adjustment to Model Heads from Raw Video Sequences. In *International Conference on Computer Vision*, Corfu, Greece, September 1999.
13. D. Garcès-Casao. Body silhouette extraction from video sequences. Technical report, Computer Graphics Lab, EPFL, http://ligwww.epfl.ch/∼plaenker/papers/garces.pdf, 1999.
14. D.M. Gavrila. The visual analysis of human movement: A survey. *Computer Vision and Image Understanding*, 73(1), 1999.
15. D.M. Gavrila and L. Davis. 3d model-based tracking of humans in action : A multi-view approach. In *Conference on Computer Vision and Pattern Recognition*, pages 73–80, San Francisco, CA, June 1996.
16. A. Gelb. *Applied Optimal Estimation.* MIT Press, Cambridge, MA, 1974.
17. L. Herda, P. Fua, R. Plänkers, R. Boulic, and D. Thalmann. Skeleton-Based Motion Capture for Robust Reconstruction of Human Motion. In *Computer Animation*, Philadelphia, USA, May 2000. to appear.
18. A. Hilton, D. Beresford, T. Gentils, R. Smith, and W. Sun. Virtual People: Capturing Human Models to Populate Virtual Worlds. In *Computer Animation*, Geneva, Switzerland, May 1999.
19. M. Isard and A. Blake. CONDENSATION - conditional density popagation for visual tracking. *International Journal of Computer Vision*, 29(1):5–28, August 1998.
20. I.A. Kakadiaris and D. Metaxas. 3D Human body model acquisition from multiple views. *International Journal of Computer Vision*, 30(3):191–218, December 1998.
21. I.A. Kakadiaris, D. Metaxas, and R. Bajcsy. Inferring 2D object structure from the deformation of apparent contours. *Computer Vision and Image Understanding*, 65(2):129–147, February 1997.
22. K. Konolige. Small Vision Systems: Hardware and Implementation. In *Eighth International Symposium on Robotics Research*, Hayama, Japan, October 1997.
23. F. Lerasle, G. Rives, M. Dhome, and A. Yassine. Human Body Tracking by Monocular Vision. In *European Conference on Computer Vision*, pages 518–527, Cambridge, England, April 1996.
24. T.B. Moeslund and E. Granum. A Survey of Computer Vision-Based Human Motion Capture. *Computer Vision and Image Understanding*, 2001. to appear.
25. E.N. Mortensen and W.A. Barrett. Intelligent Scissors for Image Composition. In *Computer Graphics, SIGGRAPH Proceedings*, pages 191–198, Los Angeles, CA, August 1995.
26. W.H. Press, B.P. Flannery, S.A. Teukolsky, and W.T. Vetterling. *Numerical Recipes, the Art of Scientific Computing.* Cambridge U. Press, Cambridge, MA, 1986.
27. J. Shen and D. Thalmann. Interactive shape design using metaballs and splines. In *Implicit Surfaces*, Grenoble, France, April 1995.
28. M.-C. Silaghi, R. Plänkers, R. Boulic, P. Fua, and D. Thalmann. Local and global skeleton fitting techniques for optical motion capture. In *Workshop on Modelling and Motion Capture Techniques for Virtual Environments*, Geneva, Switzerland, November 1998.
29. St. Sullivan, L. Sandford, and J. Ponce. Using geometric distance fits for 3–d. object modeling and recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 16(12):1183–1196, December 1994.
30. W. Sun, A. Hilton, R. Smith, and J. Illingworth. Layered animation of captured data. *The Visual Computer*, 2000. (*to appear*).
31. C.J. Taylor. Reconstructing articulated objects from point correspondences in a single uncalibrated image. In submission, 1999.
32. D. Thalmann, J. Shen, and E. Chauvineau. Fast Realistic Human Body Deformations for Animation and VR Applications. In *Computer Graphics International*, Pohang,Korea, June 1996.
33. R. Vaillant and O.D. Faugeras. Using Occluding Contours for 3D Object Modeling. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, February 1992.